Fullscreen XNA games on Windows Phone 8 devices

This article explains how to create full screen XNA apps that will run correctly on all Windows Phone 8 devices.

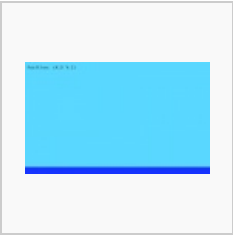
31 Mar 2013

Introduction

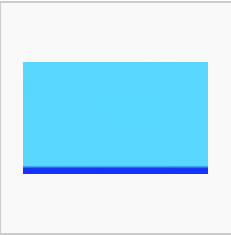
Microsoft promised that XNA applications built for Windows Phone 7 could be run on Windows Phone 8 devices without recompiling. While binary compatibility is one thing, it doesn't mean that everything will work out of the box. One key issue is the addition of new resolutions for Windows Phone 8 devices. While developers could previously count on only one target resolution - 800x480 pixels, Windows Phone 8 devices come in two additional resolutions: 1280x768 (WXGA) and 1280x720 (also know as 720p). XNA for Windows Phone was built with one limitation - back buffer, the part of the memory that was rendered to and which was then copied on the screen, was limited to maximum resolution of 800x480. This posed no problems for Windows Phone 7 devices and poses no problem for the WVGA (800x480) Windows Phone 8 devices. Things are slightly different for the other two resolutions.

Fullscreen XNA applications actually work as expected on devices with 1280x768 resolution - they fill the whole screen. This is due to the built in hardware scaler which automatically scales up image to the real resolution. There are obviously some artifacts present due to the scaling but no other problems.

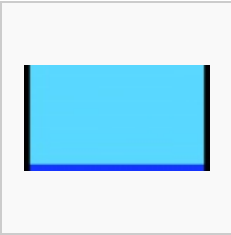
However, when running fullscreen XNA applications on 720p devices, there are black bars on either side of the screen. Since you can scale 800x480 back buffer to 1200x720 pixels, this leaves two black bars on both top and bottom of the screen. Compare the results of running the same simple applications on all three resolutions:



Simple fullscreen XNA game running on WVGA devices (800x480)



Simple fullscreen XNA game running on WXGA devices (1280x768)



Simple fullscreen XNA game running on 720p devices (1280x720)

If you target XNA, you cannot target Windows Phone 8 devices. This means that if there is a fix for this problem, it must be done using the existing API for Windows Phone 7.

Solution

We mentioned that Windows Phone devices feature hardware scaler which stretches images. The solution to this problem is to adjust the back buffer dimensions to 800x450 if we detect that the application is running on 720p devices. Why that resolution? Because if you scale it up, you get 1280x720 which means that the application will actually be fullscreen. So all we need to do is to detect this special case and then adjust the back buffer size.

First we will check if we are running on Windows Phone 8 device. This can be done using `Environment.OSVersion` property:

```
if (Environment.OSVersion.Version.Major == 8)
{

}
```

Now that we are sure that we are running on Windows Phone 8 device, we need to check if we are on 720p device, for other resolutions we don't have to do anything. We can find out the aspect ration for the device is by `Application.Current.Host.Content.ScaleFactor` property. When you create a new project from the Windows Phone Game template, you won't have Application class. For that we need to add reference to System.Windows first. But since we are targeting Windows Phone 7 devices, you will quickly find out that the desired property, `ScaleFactor` does not exist. It was introduced in Windows Phone 8 API which we cannot access. At least directly.

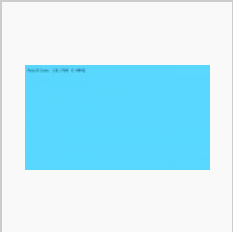
(C) Copyright Nokia 2014. All rights reserved.

There is one way to get the value of `ScaleFactor` property - by using reflection:

```
int? scaleFactor = null;
var content = Application.Current.Host.Content;
var scaleFactorProperty = content.GetType().GetProperty("ScaleFactor");
if (scaleFactorProperty != null)
{
  scaleFactor = scaleFactorProperty.GetValue(content, null) as int?;
}
if (scaleFactor == null)
  scaleFactor = 100;

if (scaleFactor == 150)
{
  graphics.PreferredBackBufferHeight = 450;
  graphics.PreferredBackBufferWidth = 800;
}
```

The value for `ScaleFactor` will be 150 for 720p devices, 160 for devices with resolution 1280x768 and 100 for devices with resolution 800x480. After applying all changes and running the application in either 720p emulator or on a 720p device, the result will be similar to the following image:



No black bars when running the application on 720 devices with resized back buffer,

What about touch?

When you query touch input, the locations of touch points will never exceed back buffer bounds. This means that the lower right end of the screen (when held in landscape mode) will return 799x449 on 720p devices with the above trick. This is really important to keep in mind and you need to perform extensive testing to ensure that application runs without problems.

Since 720p displays are a little bit wider than the other two display types, you must also alter your artwork. Instead of cutting 30 pixels from the bottom of the screen (or removing in total 30 pixels both from top and bottom of the screen), consider extending the artwork to the sides. Notice that 750x450 is scaled down 800x480. You can pretend that your game is still in the original bounds if the logic demands it and simply pad the center content with the left and right extra graphics without any game logic.

But different applications may require different handling. Ensure that your application is properly tested and that it looks as expected in this case.

Summary

Supporting multiple devices is always a challenge, even without API changes. When your application can be run on devices with different resolutions and aspect rations, you need to address that in a special way. In some cases it will be relatively straightforward, but in some cases extra code is necessary.

You can download the sample solution from the following link: File:XNAOnWP8.zip

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.